オペレーションズリサーチ 2010 (6)

6 組合せ最適化

6.1 組合せ最適化と整数計画問題

組合せ最適化問題 (combinatorial optimization problem): 組合せ的な性質を含む最適化問題である．ネットワーク最適化問題には、組合せ最適化問題とみなせるものが多い。また、組合せ最適化問題には、整数計画問題として定式化できるものが多い．

整数計画問題 (IP: integer programming): 一部 (または全て) の変数の取りうる値が整数値に限定されている数理計画問題である．整数変数と連続変数の両方を持つ場合には，**混合整数計画問題** (MIP: mixed integer programming) とも呼ぶ．

0-1 計画問題 (0-1 programming problem): 一部 (または全て) の変数のとりうる値が 0 または 1 に限定されている数理計画問題

6.2 整数計画問題と線形計画問題

整数 (0-1) 計画問題の例題

$$\begin{array}{lll} (P) & \max & z = 7x_1 + 8x_2 + 3x_3 \\ & \text{s. t.} & 3x_1 + 4x_2 + 2x_3 \leq 6 \\ & & x_1, x_2, x_3 \in \{0, 1\} \end{array}$$

とその整数条件を緩和した線形計画問題

$$\begin{array}{lll} (\bar{P}) & \max & z = 7\bar{x}_1 + 8\bar{x}_2 + 3\bar{x}_3 \\ & \text{s. t.} & 3\bar{x}_1 + 4\bar{x}_2 + 2\bar{x}_3 \leq 6 \\ & & \bar{x}_1, \bar{x}_2, \bar{x}_3 \in [0, 1] \end{array}$$

を考える．このように制約条件を緩めた問題を**緩和問題** (relaxation problem) という．

- 整数計画問題 (P) の最適解を x^*，そのときの最適値を z^* とし，線形計画問題 $(\bar{P})$ の最適解を $\bar{x}^*$，そのときの最適値を $\bar{z}^*$ とする．x^* は問題 $(\bar{P})$ の実行可能解であるから，

  $$z^* \leq \bar{z}^*$$

  が成立する．すなわち問題 $(\bar{P})$ の最適値は，問題 (P) の最適値の上界となっている．(一般に，最大化問題とその制約条件を緩和した問題の間でこの性質が成り立つ)

- 線形計画問題 $(\bar{P})$ の最適解がすべて整数ならば，それは整数計画問題 (P) の最適解である．(一般に，緩和問題の最適解がもとの問題の制約をすべてみたすならば，もとの問題の最適解である)

6.3 ナップザック問題 (knapsack problem)

- **問題：**鈴木君は、山のふもとから、山頂にある店まで荷物運びのアルバイトをすることにした。荷物は、A, B, C, D の 4 品あり、それぞれ重さが 5kg, 4kg, 3kg, 2kg である。また、それぞれの荷物を運んだときの手当は、順に 300 円, 240 円, 150 円, 80 円であるという。鈴木君は、荷物を運ぶのにナップザックを利用するが、7kg より重い荷物を入れると破れてしまう。さて、どの荷物を運べば手当の総額 (利益) が最も多くなるか。

- **モデル化：**まず、0 または 1 という値をとる 4 つの 0-1 変数 x_A, x_B, x_C, x_D を用意する。これらの変数は品物をナップザックに入れて運ぶとき 1、運ばないとき 0 という値をとるものとする。このとき問題は、次のように定式化できる。

$$\begin{array}{ll} \text{最大化} & 300x_A + 240x_B + 150x_C + 80x_D \\ \text{制約条件} & 5x_A + 4x_B + 3x_C + 2x_D \leq 7 \\ & x_A, x_B, x_C, x_D \in \{0, 1\} \end{array}$$

- 一般に n 個の荷物がある場合のナップザック問題は、次のように定式化できる。

$$\begin{array}{lll} (KP) & \max & z = \sum_{j=1}^{n} c_j x_j \\ & \text{s. t.} & \sum_{j=1}^{n} a_j x_j \leq b \\ & & x_j \in \{0, 1\}, j = 1, 2, \cdots, n \end{array}$$

ただし，すべての係数 $a_j, c_j (j = 1, 2, \cdots, n)$ と b は正の値である．
また，次の仮定をおく．

(1) $\sum_{j=1}^{n} a_j > b$

(2) $\frac{c_1}{a_1} \geq \frac{c_2}{a_2} \geq \cdots \geq \frac{c_n}{a_n}$

ナップザック問題を解くアルゴリズムには次のようなものがある。

- **貪欲アルゴリズム：**近似解を求めるアルゴリズムである。

- **完全列挙法：**すべての可能な運び方を列挙し、その中で制約条件をみたし利益が最大となるものを見つける方法である。品物の数 n が大きくなると、すべての可能な運び方の数が爆発的に（指数オーダで）増える。

- **分枝限定法：**完全列挙を基本とするが、明らかに最適な解が得られない場合を列挙せずに済ますことにより、効率を上げる解法である。実際に大規模な問題を解くことが可能な場合が多い。

- **動的計画法：**荷物がひとつもない場合からはじめ、荷物の数を一つずつ増やして順に考える解法である。すべての品物の重さが整数値であり、総重量 b の値があまり大きくないときに有効な解法である。

6.4 ナップザック問題を解く分枝限定法 (branch–and–bound method)

(a) 上界と下界

- ナップザック問題の緩和問題を考える

$$(\overline{KP}) \quad \begin{array}{ll} \max & \bar{z} = \sum_{j=1}^{n} c_j \bar{x}_j \\ \text{s. t.} & \sum_{j=1}^{n} a_j \bar{x}_j \leq b \\ & 0 \leq \bar{x}_j \leq 1, j = 1, 2, \cdots, n \end{array}$$

- この問題の最適解は次のようにして簡単に得られる．不等式

$$\sum_{j=1}^{P-1} a_j \leq b < \sum_{j=1}^{P} a_j$$

をみたす添え字 p が一意に定まる．このとき

$$\bar{x}_j^* = \begin{cases} 1 & j = 1, 2, \cdots, p-1 \\ \frac{b - \sum_{i=1}^{p-1} a_i}{a_p} & j = p \\ 0 & j = p+1, \cdots, n \end{cases}$$

とした解 $\bar{x}^* = (\bar{x}_1^*, \bar{x}_2^*, \cdots \bar{x}_n^*)$ が問題 $(\overline{KP})$ の最適解である．このときの最適値を $\bar{z}^* = \sum_{j=1}^{n} c_j \bar{x}_j^*$ とする．

- もし $\bar{x}_p^*$ が整数ならば，$\bar{x}^*$ は問題 (KP) の最適解である．そうでない場合にも，$\bar{z}^*$ が問題 (KP) の最適値の上界となる．
- 上記の解において x_p を 0 とした解を $\tilde{x}$ とする．

$$\tilde{x}_j = \begin{cases} 1 & j = 1, 2, \cdots, p-1 \\ 0 & j = p, p+1, \cdots, n \end{cases}$$

（この解 $\tilde{x}$ が貪欲アルゴリズムによる近似解である。）このときの目的関数値を $\tilde{z} = \sum_{j=1}^{n} c_j \tilde{x}_j$ とする．$\tilde{x}$ は問題 (KP) の実行可能解であるから，最適値 z^* の下界となる．
以上の議論から，(KP) の最適値 z^* は不等式

$$\tilde{z} \leq z^* \leq \bar{z}^*$$

をみたす．

(b) 分枝操作: ナップザック問題 (KP) において，1 つの適当な変数 x_s(分枝変数とよぶ) を選び，その変数の値を 0 に固定した場合と，1 に固定した場合の 2 つの問題 (子問題とよぶ) を考える

$$(P_1) \quad \begin{array}{ll} \max & z_1 = \sum_{j \neq s} c_j x_j \\ \text{s. t.} & \sum_{j \neq s} a_j x_j \leq b \\ & x_j \in \{0, 1\}, j = 1, 2, \cdots, s-1, s+1, \cdots, n \end{array}$$

$$(P_2)\quad \begin{array}{ll} \max & z_2 = \sum_{j\neq s} c_j x_j + c_s \\ \text{s. t.} & \sum_{j\neq s} a_j x_j \leq b - a_s \\ & x_j \in \{0,1\}, j = 1,2,\cdots,s-1,s+1,\cdots,n \end{array}$$

子問題 (P_1) と (P_2) を解き，それぞれの最適解と最適値を求めれば，最適値の大きい方の解が問題 (KP) の最適解となる．
このように 1 つの親問題の実行可能領域をいくつかに分割し，複数の子問題を生成することを分枝操作という．

- 子問題 (P_1) においても，x_s とは別の変数 x_t を分枝変数として分枝操作を施せば，(P_1) を親問題とする 2 つの子問題が得られる．この操作を可能な限り繰り返せば，(図のような) 列挙木が得られる．
- 列挙木の末端には，すべての変数が 0 か 1 に固定された自明な解をもつ 2^n 個の子問題が位置する．

(c) 限定操作

- 分枝操作の途中で得られる子問題 (P_k) において，次のいずれかの場合には，その問題をさらに分枝する必要がない．

  (i) 子問題 (P_k) に実行可能解が存在しない．

  (ii) 子問題 (P_k) の最適解が得られる．

  (iii) 子問題 (P_k) に問題 (KP) の最適解が含まれないことが判明する．

  このような場合に子問題の分枝を止めることを限定操作という．
- ナップザック問題 (KP) の実行可能解 x^0 とそのときの目的関数値 z^0 が得られているものとする．子問題 (P_k) について，その整数条件を緩和した線形計画問題を解けば，子問題 (P_k) の最適解が得られるか，あるいは最適値 z_k^* の下界 $\tilde{z}_k$ と上界 $\bar{z}_k^*$ を得ることができる．
  後者の場合には，もし
  $$\bar{z}_k^* < z^0$$
  が成立すれば，上述の (iii) の場合となる．($\bar{z}_k^* = z^0$ の場合にも，子問題 (P_k) には z^0 よりも大きな目的関数値を達成する解が存在しないので，さらに分枝する必要はない．)
- また，$z^0 < \tilde{z}_k$ が成立する場合には，z^0 よりも大きな目的関数値を達成する実行可能解を得ることができる．

(d) 分枝限定法

ステップ 1： 必要ならば変数の順を入れ換え，仮定をみたすようにする．

ステップ 2： $N = \{(KP)\}, k = 0, z^0 = -\infty$ とおく．

ステップ 3： $N = \phi$ ならば終了．
そうでなければ，問題 $(P) \in N$ を一つ選び，それを N から除く．

ステップ 4： 問題 (P) の緩和問題 $(\bar{P})$ に実行可能解がなければステップ 3 に戻る．問題 $(\bar{P})$ に可能解があれば，その最適解 $\bar{x}^*$ と最適値 $\bar{z}^*$ を求める．

ステップ 5： $\bar{z}^* \leq z^0$ であればステップ 3 に戻る．

ステップ 6： $\bar{x}^*$ が整数解ならば $x^0 \leftarrow \bar{x}^*$，$z^0 \leftarrow \bar{z}^*$ としてステップ 3 に戻る．

ステップ 7： $\bar{x}^*$ を整数解に切り下げ，(P) の実行可能解 $\tilde{x}$ とそのときの目的関数値 $\tilde{z}$ を求める．
$\tilde{z} > z^0$ ならば $x^0 \leftarrow \tilde{x}$，$z^0 \leftarrow \tilde{z}$ とする．

ステップ 8： 分枝変数 x_s を定め，(P) において x_s をそれぞれ 0 と 1 に固定した子問題 (P_{k+1}) と (P_{k+2}) を生成する．$N \leftarrow N \cup \{(P_{k+1}), (P_{k+2})\}$, $k \leftarrow k+2$ としてステップ 3 に戻る．

- ステップ 3 での問題 (P) の選び方
  - 最良上界規則
  - 幅優先規則
  - 深さ優先規則
- ステップ 8 での変数 x_s の選び方
  - 緩和問題の最適解で非整数値をとる変数
  - 値が固定されていない最小添字の変数

6.4.1 問題

次のナップザック問題について，問 (1) と (2) に答えなさい．

$$\begin{array}{lll} \text{最大化} & z = & 7x_1 + 6x_2 + 9x_3 + 2x_4 + x_5 \\ \text{制約} & & 4x_1 + 2x_2 + 8x_3 + 3x_4 + 4x_5 \leq 10 \\ & & x_1, x_2, x_3, x_4, x_5 \in \{0, 1\} \end{array}$$

(1) 条件 $x_1, x_2, x_3, x_4, x_5 \in \{0, 1\}$ を条件 $x_1, x_2, x_3, x_4, x_5 \in [0, 1]$ に変更した緩和問題の最適解を求めなさい．

(2) 分枝限定法を使いナップザック問題の最適解を求めなさい..

6.5 巡回セールスマン問題

- 有向ネットワーク $N = \{V, E, c\}$
  頂点　$V = \{1, 2, \cdots, n\}$
  枝　$E \subset V \times V$
  長さ (距離)　$c_{ij} \geq 0 \ \ \forall (i, j) \in E$

- (ネットワークから n 個の枝を選び) 頂点 1 から出発し，すべての頂点を通ってもとの頂点 1 へ戻る巡回路のうち，長さが最小となるものを求める．

- 変数

$$x_{ij} = \begin{cases} 1 & \text{枝 } (i,j) \text{ を巡回路に使う} \\ 0 & \text{枝 } (i,j) \text{ を巡回路に使わない} \end{cases}$$

- 定式化

$$\begin{array}{ll} \min & z = \sum_{(i,j)\in E} c_{ij}x_{ij} \\ \text{s. t.} & \sum_{(i,j)\in E} x_{ij} = 1 \text{ for } i = 1,2,\cdots n, \\ & \sum_{(i,j)\in E} x_{ij} = 1 \text{ for } j = 1,2,\cdots n, \\ & x_{ij} \in \{0,1\} \text{ for } (i,j) \in E \\ & \{(i,j) \in E : x_{ij} = 1\} \text{ が部分巡回路を含まない．} \end{array}$$

- 緩和問題
  部分巡回路を含まないという条件を除く．

$$\downarrow$$

  整数計画問題であるが，LP を解けば最適解が得られる．

- 子問題: (ある問題において，枝 (i,j) を選び，その枝を必ず使う場合と使わない場合に分ける.)
  一般に，子問題では，使ってはいけない枝の集合 $E^0 \subset E$ と必ず使う枝の集合 $E^1 \subset E$ が定められている．この子問題を $TSP[E^0, E^1]$ と表記する．

- 分枝操作: $TSP[E^0, E^1]$ と枝 $(i,j) \in E - (E^0 \cup E^1)$ に対し，2 つの子問題

$$TSP[E^0 \cup (i,j), E^1]$$
$$TSP[E^0 \cup (j,i), E^1 \cup (i,j)]$$

  を考える．